CLAIMS

What is claimed is:

1. A method comprising:

receiving a first code segment, the first code segment having a set of instances of a parallel construct, each of the set of instances of the parallel construct comprised of a task; and

translating the first code segment to a second code segment, the second code segment, when being executed to perform operations comprising:

allocating a shared value, the shared value to indicate a most current one of the set of instances encountered by one of a team of threads,

allocating a private value for each of the team of threads, the private value to indicate one of the set of instances encountered by the private value's corresponding thread of the team of threads,

maintaining the shared valuewith the team of threads, and

maintaining the private value of each of the team of threads with the private value's corresponding thread of the team of threads.

2. The method of claim 1 wherein the thread undertakes the task of one of the set of instances if the shared value and the private value indicate a same one of the set of instances.

3. The method of claim 1 wherein maintaining the shared value comprises incrementing the shared value when one of the team of threads undertakes the task of one of the set of instances.

4. The method of claim 1 wherein maintaining the private value comprises incrementing the private value when the private value's corresponding thread of the team of threads encounters one of the set of instances.

5. The method of claim 1 further comprising the thread maintaining a second private value, the second private value being a boundary for accessing the shared value.

6. A method comprising:

receiving a first code segment, the first code segment having a set of instances of a parallel construct, each of the set of instances being associated with a task; and

for each of the set of instances, generating a second code segment, which when executed, cause a first thread to perform operations comprising:

encountering one of the set of instances,

determining if the task of the one of the set of instances has been undertaken by a second thread,

upon determining the task has not been undertaken by the second thread, undertaking the task and indicating with a shared value the undertaking of the task,

indicating with a first private value of the first thread the one of the set of instances, and

maintaining a second private value of the first thread, the second private value being a boundary for accessing the shared value.

7. The method of claim 6 wherein determining if the task of the one of the set of instances has been undertaken by the second thread comprises the first thread comparing the shared value and its first private value.

8. The method of claim 6 wherein determining if the task of the one of the set of instances has been undertaken by the second thread comprises the first thread comparing the first private value of the first thread and the second private value of the first thread.

9. The method of claim 6 further comprising the second thread maintaining a third private value and a fourth private value, the third private value indicating a second one of the set of instances encountered by the second thread and the fourth private value indicating the second thread's boundary for accessing the shared value.

10. An apparatus comprising:

a memory to host a code segment, the code segment having a set of instances of a parallel construct, each of the set of instances of the parallel construct associated with a task; and

a set of processors coupled to the memory, the set of processors to process the code segment, to encounter one of the set of instances while processing the code segment, to generate a second code segment corresponding to the one of the set of instances, the second code segment to cause each of a team of threads to perform the following:

to determine if the task associated with the one of the set of instances has been undertaken by another one of the set of threads,

if the task has not been undertaken by another one of the set of threads, to undertake the task and to indicate with a shared value the undertaking of the task, and

to indicate with a private value the one of the set of instances.

11. The apparatus of claim 10 wherein the set of processors to determine if the task has been undertaken by another one of the set of threads comprises to compare the private value and the shared value.

12. The apparatus of claim 10 further comprising the set of processors to maintain a second private value, the second private value being a boundary to access the shared value.

13. A system comprising:

a translating unit to translate a code, the code having an instance of a parallel construct, the instance associated with a task, the translating unit to generate a code segment, the code segment to cause each of a set of threads to perform the following:

to determine if the task associated with the instance has been undertaken by another one of the set of threads,

if the task has not been undertaken by another one of the set of threads, to undertake the task and to indicate with a shared value the undertaking of the task, and

to indicate with a private value the instance;

a linker unit coupled to the translating unit, the linker unit to link the translated code with a library; and

a set of processors coupled to the translating unit and the linker unit, the set of processors to host the set of threads.

14. The system of claim 13 wherein to determine if the task associated with the instance has been undertaken by another one of the set of threads comprises to compare the private value with the shared value.

15. The system of claim 13 wherein the code segment to cause each of the set of threads to perform operations further comprising to maintain a second private value, the second private value being a boundary for the second private value's corresponding one of the set of threads to access the shared value.

16. A machine-readable medium that provides instructions, which when executed by a set of one or more processors, cause said set of processors to perform operations comprising:

receiving a first code segment, the first code segment having a set of instances of a parallel construct, each of the set of instances of the parallel construct comprised of a task; and

translating the first code segment to a second code segment, the second code segment, when being executed to perform operations comprising:

allocating a shared value, the shared value to indicate a most current one of the set of instances encountered by one of a team of threads,

allocating a private value for each of the team of threads, the private value to indicate one of the set of instances encountered by the private value's corresponding thread of the team of threads,

maintaining the shared value with the team of threads, and

maintaining the private value of each of the team of threads with the private value's corresponding thread of the team of threads.

17. The machine-readable medium of claim 16 wherein the thread undertakes the task of one of the set of instances if the shared value and the private value indicate a same one of the set of instances.

18. The machine-readable medium of claim 16 wherein maintaining the shared value comprises incrementing the shared value when one of the team of threads undertakes the task of one of the set of instances.

19. The machine-readable medium of claim 16 wherein maintaining the private value comprises incrementing the private value when the private value's corresponding thread of the team of threads encounters one of the set of instances.

20. The machine-readable medium of claim 16 further comprising the thread maintaining a second private value, the second private value being a boundary for accessing the shared value.

21. A machine-readable medium that provides instructions, which when executed by a set of one or more processors, cause said set of processors to perform operations comprising:

receiving a first code segment, the first code segment having a set of instances of a parallel construct, each of the set of instances being associated with a task; and

for each of the set of instances, generating a second code segment, which when executed, cause a first thread to perform operations comprising:

encountering one of the set of instances,

determining if the task of the one of the set of instances has been undertaken by a second thread,

upon determining the task has not been undertaken by the second thread, undertaking the task and indicating with a shared value the undertaking of the task,

indicating with a first private value of the first thread the one of the set of instances, and

maintaining a second private value of the first thread, the second private value being a boundary for accessing the shared value.

22. The machine-readable medium of claim 21 wherein determining if the task of the one of the set of instances has been undertaken by the second thread comprises the first thread comparing the shared value and its first private value.

23. The machine-readable medium of claim 21 wherein determining if the task of the one of the set of instances has been undertaken by the second thread comprises the first thread comparing the first private value of the first thread and the second private value of the first thread.

24. The machine-readable medium of claim 21 further comprising the second thread maintaining a third private value and a fourth private value, the third private value indicating a second one of the set of instances encountered by the second thread and the fourth private value indicating the second thread's boundary for accessing the shared value.